

In the United States Patent and Trademark Office

In re the application of: Brodsky)		
)		
Filed: 12/14/2000)	Group Art Unit:	2153
)		
For: Method Apparatus and)	Examiner:	Yasin M. Barqadle
Computer Program Product)		
to Crawl a Web Site)		
)		
Application No. 09/736,349)		
)		
Appellant's Docket:)		
AUS920000510US1)		

THIRD AMENDED APPEAL BRIEF

This Amended Appeal Brief is responsive to a May 20, 2009, Notification of Non-Compliant Appeal Brief (the "Notice"), which implements a May 8, 2009, Order Returning Undocketed Appeal to Examiner (the "Order") by the Board of Patent Appeals and Interferences (the "Board"), which vacates the Examiner's Answer of March 21, 2008. On page 3, the Order states that the Examiner's Answer of March 21, 2008, sets out a new ground of rejection without a required heading and/or does not include the approval of the TC Director or his/her designee. The Order goes on to describe, as a specific error in this regard, that claims 7, 15 and 23 are rejected in the Examiner's Answer.¹

The Order also states the following on page 2:

The "Summary of claimed subject matter" appearing on pages 4-8 of the Appeal Brief filed December 17, 2007 is deficient because it does not separately map independent claims 7, 15 and 23 to the specification. In addition, dependent claims 5, 6, 19, 21 and 22, which contain step plus functions, must also be separately mapped to the specification. Correction is required.

¹ Appellant's attorney discussed the situation with Examiner Barqadle on June 15, 2009. In that discussion the Examiner confirmed that he did not intend for the Examiner's Answer to convey that claims 7, 15 and 23 were rejected.

The “Summary of Claimed Subject Matter” is a section in appeal brief that requires “A concise explanation of the subject matter defined in each of the independent claims involved in the appeal, which must refer to the specification by page and line number, and to the drawing, if any, by reference characters.” MPEP 1205.02 Appellant submits that the Second Amended Appeal Brief was *not* deficient for failing to summarize claims that stood allowed at the time of its filing, i.e., claims 7, 15 and 23, since these claims are not involved in the appeal and are not the subject of review by the Board. The Order vacating the Examiner’s Answer of March 21, 2008, removes any doubt that claims 7, 15 and 23 continue to stand allowed.

The “Summary of Claimed Subject Matter” section further requires that “. . . for each dependent claim argued separately under the provisions of 37 CFR 41.37(c)(1)(vii), every means plus function and step plus function as permitted by 35 U.S.C.112, sixth paragraph, must be identified and the structure, material, or acts described in the specification as corresponding to each claimed function must be set forth with reference to the specification by page and line number, and to the drawing, if any, by reference characters.” MPEP 1205.02 Appellant submits that the Second Amended Appeal Brief was likewise *not* deficient for failing to summarize dependent claims 5, 6, 19, 21 and 22, since they were not argued separately.

Nevertheless, in order to fully cooperate, Appellant herein submits a Summary of Claimed Subject Matter that adds a separate mapping to the specification of the claims recited in the Notice, i.e., claims 5, 6, 19, 21 and 22, and also independent claims 7, 15 and 23.

Please substitute this entire Third Amended Appeal Brief for Appellant’s Second Amended Appeal Brief.

REAL PARTY IN INTEREST

The assignee, International Business Machines Corporation, is the real party in interest.

RELATED APPEALS AND INTERFERENCES

This is the first appeal (reinstated) in the present patent application. There are no other appeals or interferences known to the appellant or its legal representative. International Business Machines Corporation is the sole assignee of the patent application.

STATUS OF CLAIMS

Claims 1-7, 9-15, and 17-23 are pending in the application. The most recent Office action dealing with the substance of the claims, which is dated September 7, 2006, rejects claims 1, 2, 9, 10, 17, and 18, indicates claims 4, 12, and 20 are allowable if presented in independent form incorporating all features of their base claims, **and allows claims 7, 15, and 23.**

The claims appealed herein include claims 1, 2, 9, 10, 17, and 18.

(Claims 7, 15 and 23 remain **allowed**, but pending. Claims 3-6, 11-14 and 19-22 also remain, but are unallowed. Claims 3-6, 11-14 and 19-22 depend on one or more of claims 1, 2, 9, 10, 17, and 18. Appellant does not separately argue any of claims 3-6, 11-14 or 19-22, but submits that they are allowable at least because they depend on allowable claims. MPEP 2143.03 (citing *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988). If the Examiner or the Board of Patent Appeals and Interferences considers that any of claims 3-7, 11-15 or 19-23 must be herein declared as “appealed” in order for this Brief to be considered in proper form or in order for these claims to remain pending, then Appellant also hereby declares claims 3-7, 11-15 and 19-23 to be herein appealed.)

A first Office action, mailed July 19, 2004, rejected all claims and relied on three references, U.S. Patent No. 6,301,614 (“Najork”), U.S. Patent No. 6,026,413 (“Challenger”), and U.S. Patent No. 6,748,418 (“Yoshida”). Reply A was filed October 18, 2004, amending Claims 1, 2, 4, 5, 7, 9, 10, 12, 13, 15, 17, 18, 20, 21, and 23. Claims 8, 16, and 24 were canceled.

A second, final Office action of February 24, 2005, rejected all remaining claims and cited additional references, “Crawler-Friendly Web Servers” September 2000, (“Brandman”) and U.S. Patent No. 6,735,169 (“Albert”). In a first Request for Reconsideration, filed March 21, 2005, Appellant requested that the finality of the second Office action be withdrawn because Appellant was not granted an interview before the final rejection and because the statements of rejections of claims 7, 15 and 23 in the second Office action were identical repetition of the rejections of the first Office action, even though these claims were substantially amended in Reply A.

On March 22, Examiner and Attorney for Appellant discussed the application in a telephone conference.

A third, final Office action, mailed on June 22, 2005, withdrew the Brandman reference relied upon in the second Office action and finally rejected all claims in reliance upon Najork, Challenger, Albert and a new reference, U.S. Patent No. 6,665,658 (“DaCosta”). In a Statement of Disqualification of Reference and Request for Reconsideration, filed August 11, 2005, Appellant provided facts disqualifying the DaCosta reference and requested allowance.

A fourth Office action dated January 4, 2006, withdrew the DaCosta reference and, once again, finally rejected all claims, citing a new reference, US Patent 6,449,636 (“Kredo”). An appeal followed, for which an appeal brief was filed on June 2, 2006.

The Office action of September 7, 2006, reopened prosecution, **allowing claims 7, 15, and 23**, indicating claims 4, 12, and 20 to be allowable if presented in independent form, and otherwise maintaining the previous rejections stated herein above. Appellant requested reinstatement of the appeal in a notice sent by facsimile transmission to the USPTO on December 7, 2006.

STATUS OF AMENDMENTS

There are no amendments in connection with this appeal and none were submitted subsequent to the Office action that immediately preceded the appeal. Accordingly, the claims in the Claim Appendix herein set out the claims that are the subject of the appeal.²

SUMMARY OF CLAIMED SUBJECT MATTER

The claims in the present case relate to an embodiment of the invention in which a reference to a web page that is not simply set out on the page as a hyperlink address, but is instead specified by a script, may be produced only when a client browser executes the reference.³ Present application, page 5, lines 10 - 15. See also, page 12, line 21- page 13, line 1 (describing how the reference may be specified by a script, a selection menu, form, button or other similar element). This presents a problem for a web page crawler. According to the present application, the browser locates code for the function that is called and then executes the specified function using an applet.

Claim 1

Claim 1 describes a method for crawling a web site.

The claim has steps as follows:

First step, “querying a web site server by a crawler program, wherein at least one page of the web site has a reference, wherein the reference is specified by a script to produce an address for a next page;”

Second step, “parsing such a reference from one of the web pages by the crawler program and sending the reference to an applet running in a browser”; and

Third step, “determining the address for the next page by the browser executing the reference and sending the address to the crawler.”

The specification of the present application provides an exemplary embodiment of the invention. The specification describes querying a web site server by a crawler program, as

² The claims are as submitted in Appellant’s Reply A of October 18, 2004.

³ The broad claims in this case say a reference is *specified* by a script, because an href tag, for example, typically has a *call* to a script and not the script *itself*.

claimed. See, e.g., present application, Fig's 1 and 3, page 11, lines 9 - 17 ("Client 170 has a crawler program 171 for generating queries to the server 100 responsive to references from one web page to another in the content delivered by web server 100. Specifically, first page 140.1 is shown having a first reference 142 linking page 140.1 to page 140.2. The crawler program 171 in client 170, responsive to reference 142, queries server 100 for the second web page 140.2. Then, upon receiving web page 140.2 the client 170 queries server 100 for web page 140.3, responsive to reference 146.").

The specification further explains that at least one page of the web site has a reference, wherein the reference is specified by a script to produce an address for a next page, as claimed. See, e.g., present application, Fig's 1, 2, and 3, page 12, lines 11-13 ("In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements."); page 12, lines 11-13 ("In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements."); page 14, line 10 - page 15, line 2 (the crawler 171 is programmable to perform particular action sequences 305 for generating queries to the web server 100); page 15, lines 2-8 (at least one page of the web site has a reference specified by a script 303 to produce an address for a next page, so that the address is produced only when a client browser 205 executes the reference).

The specification further describes parsing such a reference from one of the web pages by the crawler program and sending the reference to an applet running in a browser, as claimed. See, e.g., present application, Fig. 2, page 12, lines 17- 19 ("A first web page 140.X is loaded in the browser 205. The crawler 171 includes a function 220 for parsing information from web pages loaded in the browser 205, such as 140.X, and passing the information 230 to the browser 205 for interpreting the information . . ."). See also Fig. 5 (step 510) and page 16, lines 16-17 (the crawler 171 parses a reference from one of the web pages and sends the reference to an applet running in a browser 205.).

The specification further describes determining the address for the next page by the browser executing the reference and sending the address to the crawler, as claimed. See, e.g.,

present application, Fig. 2, page 12, line 19 - page 13, line 8 (“. . . passing the information 230 to the browser 205 for interpreting the information, which concerns references to other pages, such as that from 140.X to 140.Y, and generating queries for the other pages. To understand the context of these operations, it should be understood that a reference may not be simply set out on a web page as an address, but instead may be specified by a script, selection menu, form, button or other element, so that the address needs to be produced by the client browser executing the reference. A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser. So to generate references of this sort in connection with generating the requests to the server, the crawler 171 parses each received web page, such as page 140.X, and sends references parsed from the page to a JavaScript execution engine 210 that runs in the browser 205, for generating a query for web page 140.Y. That is, browser 205 determines from the information 230 that page 140.Y is referenced in referencing information in page 140.X.”). See also Fig. 5 (steps 515 and 520); page 16, lines 17-19 and page 11, line 9 (The browser 205 determines the address for the next page by executing the parsed reference using the applet. Then the browser 205 sends the next-page address to the crawler 171, which sends it in a query to server 100).

Claim 2

Claim 2 encompasses an embodiment of claim 1. The specification explains that the browser is configured to use a certain proxy and refer to a resolver file for hostname-to-IP-address-resolution, as claimed. See, e.g., present application, Fig. 4, page 15, lines 15-17, see also Fig. 5, (steps 525 and 530) and page 16, lines 19-20 (browser 205 is configured to use a proxy 215 and refer to a resolver file 405 for hostname-to-IP-address- resolution).

The specification further explains that the web site server has an IP address and the proxy for the browser has a certain IP address, as claimed. See present application, Fig. 4, page 6, lines 8-9, page 15, lines 17-20 (the web site server 100 has an IP address and the proxy 215 for the browser 205 has a certain IP address).

The specification further explains that the resolver file indicates the certain IP address of the proxy as the IP address for the web site server, as claimed. See present application, Fig.

4, page 6, lines 9-14, page 15, line 20 - page 16, line 2 (the IP address of the proxy 215 is different than the IP address of the web site server 100, and the resolver file indicates the IP address of the proxy 215 as the IP address for the web site server 100).

Claim 5

Claim 5 encompasses an embodiment of claim 1, wherein at least one of the web pages is dynamically generated by the server responsive to corresponding ones of the queries. Claim 5 adds an additional step of processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server.

The specification explains that at least one of the web pages is dynamically generated by the server responsive to corresponding ones of the queries, as claimed. See, e.g., present application, Fig's 1, 2, 3, and 5; page 12, line 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically , such as to page 140.Y. "A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser."). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has "lists 301 and 302, for selecting parameters for generating a query," which causes the server to dynamically generate data for the page 140.X, where "... the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302 . The point to note is that obtaining the desired data requires these actions to be performed, in the proper sequence.").

The specification also describes processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, as claimed. See, e.g., present application Fig.'s 2 and 5; page 13, line 14 - page 14, line 9 ("web page 140.X . . . has operations included in it linking it to page 140.Y, that call for the server 100 to generate more dynamic data to produce web page 140.Y. Therefore, web page 140.X needs to be converted by removing these operations and replacing them with references

to a "version of" the returned web page 140.Y. The reference must be to 140.Y1, a "version of" web page 140.Y, because 140.Y may itself have operations included linking it to a next page also calling for the server 100 to generate more dynamic data to produce the next page. To deal with these issues, the crawler also includes a processing function 225. The browser 205 passes information 240 back to the crawler 171 for the crawler to use for processing the web pages and saving versions thereof. In the illustration, web pages 140.X and 140.Y are processed to generate new versions 140.X1 and 140.Y1, which are saved in storage 226.”). See also, page 16, line 23 - page 17, line 4 (“... a new version of the first web page is created, with a local file name, by replacing operations that would otherwise generate dynamic data, and linking the first page to a new version of the second page, the second page also being assigned a local file name.”).

Claim 6

Claim 6 encompasses an embodiment of claim 5, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server. Claim 6 adds the step of removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

The specification explains that at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, as claimed. See, e.g., present specification, Fig.’s 2 and 5; page 13, line 14 - page 14, line 9 (“... web page 140.X . . . has operations included in it linking it to page 140.Y, that call for the server 100 to generate more dynamic data to produce web page 140.Y. Therefore, web page 140.X needs to be converted by removing these operations and replacing them with references to a "version of" the returned web page 140.Y. The reference must be to 140.Y1, a "version of" web page 140.Y, because 140.Y may itself have operations included linking it to a next page also calling for the server 100 to generate more dynamic data to produce the next page. To deal with these issues, the crawler also includes a processing function 225. The browser 205 passes

information 240 back to the crawler 171 for the crawler to use for processing the web pages and saving versions thereof. In the illustration, web pages 140.X and 140.Y are processed to generate new versions 140.X1 and 140.Y1, which are saved in storage 226.”).

The specification also describes removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages, as claimed. See, e.g., present application, page 16, line 23 - page 17, line 4 (“... a new version of the first web page is created, with a local file name, by replacing operations that would otherwise generate dynamic data, and linking the first page to a new version of the second page, the second page also being assigned a local file name.”).

Claim 7

Claim 7 encompasses a method for reducing dynamic data generation on a web site server. The claim has steps as follows:

First Step: querying a web site server by a crawler program responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web pages being dynamically generated.

Second Step: processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, the operation including a number of non-hypertext-link elements on the first page selected in a particular sequence, and wherein the step of processing the server generated web pages comprises the step of removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

Regarding support for claim 7, the specification describes querying a web site server by a crawler program responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web pages being dynamically generated, as claimed. See, e.g., present application, Fig's 1, 2, 3, and

5; page 11, lines 9 - 17 (“Client 170 has a crawler program 171 for generating queries to the server 100 responsive to references from one web page to another in the content delivered by web server 100. Specifically, first page 140.1 is shown having a first reference 142 linking page 140.1 to page 140.2. The crawler program 171 in client 170, responsive to reference 142, queries server 100 for the second web page 140.2. Then, upon receiving web page 140.2 the client 170 queries server 100 for web page 140.3, responsive to reference 146.”); page 12, lines 11-13 (“In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements.”); page 12, lines 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically, such as to page 140.Y. “A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser.”). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has “lists 301 and 302, for selecting parameters for generating a query,” which causes the server to dynamically generate data for the page 140.X, where “. . . the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302 . The point to note is that obtaining the desired data requires these actions to be performed, in the proper sequence.”). See also, page 15, lines 2-8 (at least one page of the web site has a reference specified by a script 303 to produce an address for a next page, so that the address is produced only when a client browser 205 executes the reference); and page 16, lines 16-19 (“Next, at step 510 the page is parsed and information 230 is sent to the JEE 210. The JEE 210 and browser 205 determine an address from the information 230, in step 515, and the browser sends a query for the server 100, in step 520.”).

The specification also describes processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used

to generate further requests to the server, as claimed. See, e.g., present application, Fig.'s 2 and 5; page 13, line 14 - page 14, line 9 ("... web page 140.X ... has operations included in it linking it to page 140.Y, that call for the server 100 to generate more dynamic data to produce web page 140.Y. Therefore, web page 140.X needs to be converted by removing these operations and replacing them with references to a "version of" the returned web page 140.Y. The reference must be to 140.Y1, a "version of " web page 140.Y, because 140.Y may itself have operations included linking it to a next page also calling for the server 100 to generate more dynamic data to produce the next page. To deal with these issues, the crawler also includes a processing function 225. The browser 205 passes information 240 back to the crawler 171 for the crawler to use for processing the web pages and saving versions thereof. In the illustration, web pages 140.X and 140.Y are processed to generate new versions 140.X1 and 140.Y1, which are saved in storage 226.").

The specification also explains that the operation includes a number of non-hypertext-link elements on the first page selected in a particular sequence, as claimed. See, e.g., present application, page 12, lines 11-13 ("In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements."); page 12, lines 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically, such as to page 140.Y. "A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser."). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has "lists 301 and 302, for selecting parameters for generating a query," which causes the server to dynamically generate data for the page 140.X, where "... the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302. The point to note is that obtaining the desired data requires these actions to be performed, in the proper sequence.").

The specification also explains that processing the server generated web pages includes the step of removing the operation from the first server generated web page and

replacing the operation with a reference to a version of another of the server generated web pages, as claimed. See, e.g., present application, page 16, line 23 - page 17, line 4 (“... a new version of the first web page is created, with a local file name, by replacing operations that would otherwise generate dynamic data, and linking the first page to a new version of the second page, the second page also being assigned a local file name.”).

Claim 9

Claim 9 has language similar to claim 1 and is directed to a computer program product for crawling a web site. The specification describes a computer program product residing on a computer usable medium having computer readable program code, as claimed. See present application, page 17, lines 5-12 (the computer program product includes computer-readable storage media).

The specification further describes first instructions for querying a web site server by a crawler program, as claimed. See present application, page 7, line 22 - page 8, line 1 (instructions are stored thereon for controlling operation of a processor).

The specification describes querying a web site server by a crawler program, as claimed. See, e.g., present application, Fig's 1 and 3, page 11, lines 9 - 17 (“Client 170 has a crawler program 171 for generating queries to the server 100 responsive to references from one web page to another in the content delivered by web server 100. Specifically, first page 140.1 is shown having a first reference 142 linking page 140.1 to page 140.2. The crawler program 171 in client 170, responsive to reference 142, queries server 100 for the second web page 140.2. Then, upon receiving web page 140.2 the client 170 queries server 100 for web page 140.3, responsive to reference 146.”).

The specification further explains that at least one page of the web site has a reference, wherein the reference is specified by a script to produce an address for a next page, as claimed. See, e.g., present application, Fig's 1, 2, and 3, page 12, lines 11-13 (“In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements.”); page 12, lines 11-13 (“In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The

references could be script references, selection menus, forms, buttons or other elements.”); page 14, line 10 - page 15, line 2 (the crawler 171 is programmable to perform particular action sequences 305 for generating queries to the web server 100); page 15, lines 2-8 (at least one page of the web site has a reference specified by a script 303 to produce an address for a next page, so that the address is produced only when a client browser 205 executes the reference).

The specification further describes parsing such a reference from one of the web pages by the crawler program and sending the reference to an applet running in a browser, as claimed. See, e.g., present application, Fig. 2, page 12, lines 17- 19 (“A first web page 140.X is loaded in the browser 205. The crawler 171 includes a function 220 for parsing information from web pages loaded in the browser 205, such as 140.X, and passing the information 230 to the browser 205 for interpreting the information . . .”). See also Fig. 5 (step 510) and page 16, lines 16-17 (the crawler 171 parses a reference from one of the web pages and sends the reference to an applet running in a browser 205.).

The specification further describes determining the address for the next page by the browser executing the reference and sending the address to the crawler, as claimed. See, e.g., present application, Fig. 2, page 12, line 19 - page 13, line 8 (“. . .passing the information 230 to the browser 205 for interpreting the information, which concerns references to other pages, such as that from 140.X to 140.Y, and generating queries for the other pages. To understand the context of these operations, it should be understood that a reference may not be simply set out on a web page as an address, but instead may be specified by a script, selection menu, form, button or other element, so that the address needs to be produced by the client browser executing the reference. A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser. So to generate references of this sort in connection with generating the requests to the server, the crawler 171 parses each received web page, such as page 140.X, and sends references parsed from the page to a JavaScript execution engine 210 that runs in the browser 205, for generating a query for web page 140.Y. That is, browser 205 determines from the information 230 that page 140.Y is referenced in referencing information in page 140.X.”). See also Fig. 5 (steps 515 and 520); page 16, lines 17-19 and page 11, line

9 (The browser 205 determines the address for the next page by executing the parsed reference using the applet. Then the browser 205 sends the next-page address to the crawler 171, which sends it in a query to server 100).

Claim 10

Claim 10 encompasses an embodiment of claim 9. The specification describes, a computer program product of claim 9, in which the browser is configured to use a certain proxy and refer to a resolver file for hostname-to-IP-address-resolution, as claimed. See present application, Fig. 4, page 15, lines 15-17, see also Fig. 5 (steps 525 and 530) and page 16, lines 19-20 (the browser 205 is configured to use a proxy 215 and refer to a resolver file 405 for hostname-to-IP-address-resolution).

The specification further explains that the web site server has an IP address and the proxy for the browser has a certain IP address, as claimed. See present application, Fig. 4, page 6, lines 8-9, page 15, lines 17-20 (the web site server 100 has an IP address and the proxy 215 for the browser 205 has a certain IP address).

The specification further explains that the certain IP address of the proxy is different than the IP address of the web site server, and that the resolver file indicates the certain IP address of the proxy as the IP address for the web site server, as claimed. See present application, Fig. 4, page 6, lines 9-14, page 15, line 20 - page 16, line 2 (the IP address of the proxy 215 is different than the IP address of the web site server 100, and the resolver file indicates the IP address of the proxy 215 as the IP address for the web site server 100).

Claim 15

Claim 15 encompasses a computer program product form of the invention that has language like claim 7. Regarding support for claim 15, the specification describes a computer program product residing on a computer usable medium having computer readable program code, as claimed. See present application, page 17, lines 5-12.

The specification describes first instructions for querying a web site server by a crawler program responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web

pages being dynamically generated, as claimed. See, e.g., present application, Fig's 1, 2, 3, and 5; page 11, lines 9 - 17 ("Client 170 has a crawler program 171 for generating queries to the server 100 responsive to references from one web page to another in the content delivered by web server 100. Specifically, first page 140.1 is shown having a first reference 142 linking page 140.1 to page 140.2. The crawler program 171 in client 170, responsive to reference 142, queries server 100 for the second web page 140.2. Then, upon receiving web page 140.2 the client 170 queries server 100 for web page 140.3, responsive to reference 146."); page 12, lines 11-13 ("In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements."); page 12, lines 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically, such as to page 140.Y. "A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser."). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has "lists 301 and 302, for selecting parameters for generating a query," which causes the server to dynamically generate data for the page 140.X, where "... the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302. The point to note is that obtaining the desired data requires these actions to be performed, in the proper sequence."). See also, page 16, lines 16-19 ("Next, at step 510 the page is parsed and information 230 is sent to the JEE 210. The JEE 210 and browser 205 determine an address from the information 230, in step 515, and the browser sends a query for the server 100, in step 520.").

The specification also describes second instructions for processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, as claimed. See,

e.g., present application, Fig.'s 2 and 5; page 13, line 14 - page 14, line 9 ("... web page 140.X... has operations included in it linking it to page 140.Y, that call for the server 100 to generate more dynamic data to produce web page 140.Y. Therefore, web page 140.X needs to be converted by removing these operations and replacing them with references to a "version of" the returned web page 140.Y. The reference must be to 140.Y1, a "version of" web page 140.Y, because 140.Y may itself have operations included linking it to a next page also calling for the server 100 to generate more dynamic data to produce the next page. To deal with these issues, the crawler also includes a processing function 225. The browser 205 passes information 240 back to the crawler 171 for the crawler to use for processing the web pages and saving versions thereof. In the illustration, web pages 140.X and 140.Y are processed to generate new versions 140.X1 and 140.Y1, which are saved in storage 226.").

The specification also explains that the operation includes a number of non-hypertext-link elements on the first page selected in a particular sequence, as claimed. See, e.g., present application, page 12, lines 11-13 ("In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements."); page 12, lines 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically, such as to page 140.Y. "A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser."). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has "lists 301 and 302, for selecting parameters for generating a query," which causes the server to dynamically generate data for the page 140.X, where "... the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302. The point to note is that obtaining the desired data requires these actions to be performed, in the proper sequence.").

The specification also describes instructions for removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages, as claimed. See, e.g., present application, page 16, line 23 -

page 17, line 4 (“... a new version of the first web page is created, with a local file name, by replacing operations that would otherwise generate dynamic data, and linking the first page to a new version of the second page, the second page also being assigned a local file name.”).

Claim 17

Claim 17 encompasses an apparatus form of the invention having language similar to claim 1, including a processor connected to a network, and a storage device connected to the processor and the network, wherein the storage device is for storing a program for controlling the processor, and wherein the processor is operative with the program to execute a crawler program, as claimed. See, e.g., present application, Fig. 1; page 11, lines 2-8; page 7, lines 22-23.

The specification describes querying a web site server by a crawler program, as claimed. See, e.g., present application, Fig's 1 and 3, page 11, lines 9 - 17 (“Client 170 has a crawler program 171 for generating queries to the server 100 responsive to references from one web page to another in the content delivered by web server 100. Specifically, first page 140.1 is shown having a first reference 142 linking page 140.1 to page 140.2. The crawler program 171 in client 170, responsive to reference 142, queries server 100 for the second web page 140.2. Then, upon receiving web page 140.2 the client 170 queries server 100 for web page 140.3, responsive to reference 146.”).

The specification further explains that at least one page of the web site has a reference, wherein the reference is specified by a script to produce an address for a next page, as claimed. See, e.g., present application, Fig's 1, 2, and 3, page 12, lines 11-13 (“In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements.”); page 12, lines 11-13 (“In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements.”); page 14, line 10 - page 15, line 2 (the crawler 171 is programmable to perform particular action sequences 305 for generating queries to the web server 100); page 15, lines 2-8 (at least one page of the web site has a reference specified by a script 303 to produce an address for a

next page, so that the address is produced only when a client browser 205 executes the reference).

The specification further describes parsing such a reference from one of the web pages by the crawler program and sending the reference to an applet running in a browser, as claimed. See, e.g., present application, Fig. 2, page 12, lines 17- 19 (“A first web page 140.X is loaded in the browser 205. The crawler 171 includes a function 220 for parsing information from web pages loaded in the browser 205, such as 140.X, and passing the information 230 to the browser 205 for interpreting the information . . .”). See also Fig. 5 (step 510) and page 16, lines 16-17 (the crawler 171 parses a reference from one of the web pages and sends the reference to an applet running in a browser 205.).

The specification further describes determining the address for the next page by the browser executing the reference and sending the address to the crawler, as claimed. See, e.g., present application, Fig. 2, page 12, line 19 - page 13, line 8 (“. . .passing the information 230 to the browser 205 for interpreting the information, which concerns references to other pages, such as that from 140.X to 140.Y, and generating queries for the other pages. To understand the context of these operations, it should be understood that a reference may not be simply set out on a web page as an address, but instead may be specified by a script, selection menu, form, button or other element, so that the address needs to be produced by the client browser executing the reference. A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser. So to generate references of this sort in connection with generating the requests to the server, the crawler 171 parses each received web page, such as page 140.X, and sends references parsed from the page to a JavaScript execution engine 210 that runs in the browser 205, for generating a query for web page 140.Y. That is, browser 205 determines from the information 230 that page 140.Y is referenced in referencing information in page 140.X.”). See also Fig. 5 (steps 515 and 520); page 16, lines 17-19 and page 11, line 9 (The browser 205 determines the address for the next page by executing the parsed reference using the applet. Then the browser 205 sends the next-page address to the crawler 171, which sends it in a query to server 100).

Claim 18

Claim 18 encompasses an embodiment of claim 17. The specification explains the browser is configured to use a certain proxy and refer to a resolver file for hostname-to-IP-address-resolution, as claimed. See, e.g., present application, Fig. 4, page 15, lines 15-17, see also Fig. 5 (steps 525 and 530) and page 16, lines 19-20 (the browser 205 is configured to use a proxy 215 and refer to a resolver file 405 for hostname-to-IP-address-resolution).

The specification further explains that the web site server has an IP address and the proxy for the browser has a certain IP address, as claimed. See, e.g., present application, Fig. 4, page 6, lines 8-9, page 15, lines 17-20 (the web site server 100 has an IP address and the proxy 215 for the browser 205 has a certain IP address).

The specification further explains that the certain IP address of the proxy is different than the IP address of the web site server, and the resolver file indicates the certain IP address of the proxy as the IP address for the web site server, as claimed. See, e.g., present application, Fig. 4, page 6, lines 9-14, page 15, line 20 - page 16, line 2 (the IP address of the proxy 215 is different than the IP address of the web site server 100, and the resolver file indicates the IP address of the proxy 215 as the IP address for the web site server 100).

Claim 19

Claim 19 encompasses an embodiment of claim 18, wherein an onload attribute is added to one of the web pages by the proxy, and an event handler is defined for the onload attribute to set a certain variable. The claim adds the step of polling the certain variable by the applet to determine when the page is loaded.

The specification explains that an onload attribute is added to one of the web pages by the proxy, and an event handler is defined for the onload attribute to set a certain variable, as claimed. See present application, Fig. 2; page 16, lines 5-8 ("In connection with parsing web pages, the crawler 171 needs to know when a web page being received from the server has been fully loaded by the browser 205. There is a conventional HTML document attribute that is useful for this purpose, the "onload" attribute. However, in a normal case the web pages being processed by the JEE were not produced with the JEE in mind. That is, the onload

attribute may not be included in the web pages, or may not be included in a fashion suited for use with the JEE. “).

The specification also describes polling the certain variable by the applet to determine when the page is loaded, as claimed. See present application Fig. 2; page 16, lines 9-12. (“Therefore, in another aspect as shown in FIG. 2, the proxy host 215 adds the onload attribute to each received web page which does not already have the attribute, and defines an event handler that sets a certain variable. Then, this variable is polled by the JEE to determine when the page is loaded, and the JEE 210 171 signals the crawler accordingly.”).

Claim 21

Claim 21 encompasses an embodiment of claim 17, wherein at least one of the web pages is dynamically generated by the server responsive to corresponding ones of the queries. Claim 21 adds the step of processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server. The specification describes at least one of the web pages being dynamically generated by the server responsive to corresponding ones of the queries, as claimed. See, e.g., present application, Fig’s 1, 2, 3, and 5; page 12, line 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically , such as to page 140.Y. “A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser.”). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has “lists 301 and 302, for selecting parameters for generating a query,” which causes the server to dynamically generate data for the page 140.X, where “ . . the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302 . The point to note is that obtaining the desired data requires these actions to be performed, in the proper sequence.”).

The specification also describes processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be

served in response to future queries, reducing dynamic generation of web pages by the server, as claimed. See, e.g., present application Fig.'s 2 and 5; page 13, line 14 - page 14, line 9 ("web page 140.X . . . has operations included in it linking it to page 140.Y, that call for the server 100 to generate more dynamic data to produce web page 140.Y. Therefore, web page 140.X needs to be converted by removing these operations and replacing them with references to a "version of" the returned web page 140.Y. The reference must be to 140.Y1, a "version of" web page 140.Y, because 140.Y may itself have operations included linking it to a next page also calling for the server 100 to generate more dynamic data to produce the next page. To deal with these issues, the crawler also includes a processing function 225. The browser 205 passes information 240 back to the crawler 171 for the crawler to use for processing the web pages and saving versions thereof. In the illustration, web pages 140.X and 140.Y are processed to generate new versions 140.X1 and 140.Y1, which are saved in storage 226."). See also, page 16, line 23 - page 17, line 4 (" . . . a new version of the first web page is created, with a local file name, by replacing operations that would otherwise generate dynamic data, and linking the first page to a new version of the second page, the second page also being assigned a local file name.").

Claim 22

Claim 22 encompasses an embodiment of claim 21, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server. Claim 22 adds the step of removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

The specification describes at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, as claimed. See, e.g., present specification, Fig.'s 2 and 5; page 13, line 14 - page 14, line 9 (" . . . web page 140.X . . . has operations included in it linking it to page 140.Y, that call for the server 100 to generate more dynamic data to produce web page 140.Y. Therefore, web page 140.X needs to be converted

by removing these operations and replacing them with references to a "version of" the returned web page 140.Y. The reference must be to 140.Y1, a "version of " web page 140.Y, because 140.Y may itself have operations included linking it to a next page also calling for the server 100 to generate more dynamic data to produce the next page. To deal with these issues, the crawler also includes a processing function 225. The browser 205 passes information 240 back to the crawler 171 for the crawler to use for processing the web pages and saving versions thereof. In the illustration, web pages 140.X and 140.Y are processed to generate new versions 140.X1 and 140.Y1, which are saved in storage 226.”).

The specification also describes removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages, as claimed. See, e.g., present application, page 16, line 23 - page 17, line 4 (“... a new version of the first web page is created, with a local file name, by replacing operations that would otherwise generate dynamic data, and linking the first page to a new version of the second page, the second page also being assigned a local file name.”).

Claim 23

Claim 23 encompasses an apparatus form of the invention having language similar to claim 7. The browser program and the crawler program perform steps as follows:

First Step: querying a web site server by the crawler responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web pages being dynamically generated.

Second Step: processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, the operation including a number of non-hypertext-link elements on the first page selected in a particular sequence, and wherein the step of processing the server generated web pages comprises the step of removing the operation from the first server

generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

Regarding support for claim 23, the specification describes a processor connected to a network and a storage device connected to the processor and the network, wherein the storage device is for storing a program for controlling the processor, and wherein the processor is operative with the program to execute a crawler program and a browser program, as claimed. See, e.g., present application, Fig. 1; page 11, lines 2-8; page 7, lines 22-23.

Further, the specification describes querying a web site server by a crawler program responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web pages being dynamically generated, as claimed. See, e.g., present application, Fig's 1, 2, 3, and 5; page 11, lines 9 - 17 ("Client 170 has a crawler program 171 for generating queries to the server 100 responsive to references from one web page to another in the content delivered by web server 100. Specifically, first page 140.1 is shown having a first reference 142 linking page 140.1 to page 140.2. The crawler program 171 in client 170, responsive to reference 142, queries server 100 for the second web page 140.2. Then, upon receiving web page 140.2 the client 170 queries server 100 for web page 140.3, responsive to reference 146."); page 12, lines 11-13 ("In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements."); page 12, lines 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically, such as to page 140.Y. "A reference that is generated in this manner is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser."). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has "lists 301 and 302, for selecting parameters for generating a query," which causes the server to dynamically generate data for the page 140.X, where "... the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302. The point to note is that obtaining the desired data requires these actions to be performed, in the proper

sequence.”). See also, page 15, lines 2-8 (at least one page of the web site has a reference specified by a script 303 to produce an address for a next page, so that the address is produced only when a client browser 205 executes the reference); and page 16, lines 16-19 (“Next, at step 510 the page is parsed and information 230 is sent to the JEE 210. The JEE 210 and browser 205 determine an address from the information 230, in step 515, and the browser sends a query for the server 100, in step 520.”).

The specification also describes processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, as claimed. See, e.g., present application, Fig.’s 2 and 5; page 13, line 14 - page 14, line 9 (“... web page 140.X ... has operations included in it linking it to page 140.Y, that call for the server 100 to generate more dynamic data to produce web page 140.Y. Therefore, web page 140.X needs to be converted by removing these operations and replacing them with references to a "version of" the returned web page 140.Y. The reference must be to 140.Y1, a "version of " web page 140.Y, because 140.Y may itself have operations included linking it to a next page also calling for the server 100 to generate more dynamic data to produce the next page. To deal with these issues, the crawler also includes a processing function 225. The browser 205 passes information 240 back to the crawler 171 for the crawler to use for processing the web pages and saving versions thereof. In the illustration, web pages 140.X and 140.Y are processed to generate new versions 140.X1 and 140.Y1, which are saved in storage 226.”).

The specification also explains that the operation includes a number of non-hypertext-link elements on the first page selected in a particular sequence, as claimed. See, e.g., present application, page 12, lines 11-13 (“In the above discussion, it must be noted that the references between two pages could be more than just conventional hyperlinks. The references could be script references, selection menus, forms, buttons or other elements.”); page 12, lines 17- page 13, line 8 (Page 140.X has a reference that may be generated dynamically, such as to page 140.Y. “A reference that is generated in this manner is very

dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser.”). See also, page 14, line 10 - page 15, line 8 (In another example, page 140.X has “lists 301 and 302, for selecting parameters for generating a query,” which causes the server to dynamically generate data for the page 140.X, where “ . . the user performs the following action sequence 305: 1. Select Texas as the State from list 301. 2. Select Income as the Profile from list 302. 3. Click on the create button 304, which causes a script 303 to query server 100 with a request that includes the parameters selected from the lists 301 and 302 . The point to note is that obtaining the desired data requires these actions to be performed, in the proper sequence.”).

The specification also explains that processing the server generated web pages includes the step of removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages, as claimed. See, e.g., present application, page 16, line 23 - page 17, line 4 (“ . . . a new version of the first web page is created, with a local file name, by replacing operations that would otherwise generate dynamic data, and linking the first page to a new version of the second page, the second page also being assigned a local file name.”).

GROUND OF REJECTION TO BE REVIEWED ON APPEAL

1. Is the rejection in the Final Office Action proper, wherein claims 1, 9 and 17 stand rejected under 35 U.S.C. 103(a) in view of the combination of U.S. Patent No. 6,301,614 (“Najork”) and US Patent 6,449,636 (“Kredo”)?

2. Is the rejection in the Final Office Action proper, wherein claims 2, 10 and 18 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Najork, Kredo and US Patent 6,735,169 (“Albert”)?

ARGUMENTS

1. The rejection of claims 1, 9 and 17 under 35 U.S.C. 103(a) in view of the combination of U.S. Patent No. 6,301,614 (“Najork”) and US Patent 6,449,636 (“Kredo”) is not proper because there is no evidence or suggestion in the cited art of parsing a reference from one of the web pages by a crawler program and sending the reference to an applet running in a browser nor do the references teach or suggest determining the address for the next page by the browser executing the reference and sending the address to the crawler.

The subject claims state that a page has a reference and that the reference is specified by a script for producing an address for a next page. The claims go on to say that such a reference is parsed from one of the web pages by the crawler program and sent to an applet running in a browser. Further, the claims clearly tie these aspects together by stating that the address for the next page is determined by the browser “executing” the reference and sending the address to the crawler. Appellant respectfully submits that there is no evidence or suggestion in the cited art of all the limitations of the subject claims and that the rejection is, therefore, improper. “[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness.” In re Kahn, 441 F. 3d 977, 988 (CA Fed. 2006) cited with approval in KSR Int’l v. Teleflex Inc., 127 S. Ct. 1727, 82 USPQ2d 1385 (2007). All claim limitations must be considered MPEP 2143.03 (citing In re Wilson, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970)).

There is no evidence or suggestion in the cited art of parsing a reference from one of the web pages by a crawler program and sending the reference to an applet running in a browser.

The present Office action maintains Najork teaches that a method of web crawling includes “parsing such a reference from one of the web pages by the crawler program and sending the reference to an applet running in a browser,” as in the present claim 1. (Claims 9 and 17 have similar language.) However, Najork does not teach parsing *a reference* (such as a reference having that a script for producing an address for a next page) and sending the reference *to an applet running in a browser*, as claimed. Najork teaches parsing a URL by a crawler and sending the parsed URL somewhere for storing multiple URL’s, not to an applet running in a browser.

More specifically, Najork describes the problem it addresses as follows:

A web crawler is a program that automatically finds and downloads documents from host computers in an Intranet or the world wide web . . . Before the web crawler downloads the documents associated with the newly discovered URL’s, the web crawler needs to find out whether these documents have already been downloaded . . . Thus, web crawlers need efficient data structures to keep track of downloaded documents and any discovered addresses of documents to be downloaded. Such data structures are needed to facilitate fast data checking and to avoid downloading a document multiple times.

Najork, col. 1, lines 34-61. The teachings of Najork concern “the data structures and methods used to keep track of the URL’s of documents that have already been downloaded or that have already been scheduled for downloading.” Najork, col. 4, lines 54-57.

The web crawler taught by Najork includes “threads 130 for downloading web pages from the servers 112, and processing the downloaded web pages; a main web crawler procedure 140 executed by each of the threads 130; and a URL processing procedure 142 executed by each of the threads 130 to process the URL’s identified in a downloaded web page.” Najork, col. 3, lines 31-58. Each thread executes a main web crawler procedure 140 shown in Fig. 3. Najork, col. 4, lines 58-59. The web crawler thread determines the URL of the

next document to be downloaded (step 160) and then downloads the document corresponding to the URL, and processes the document (162). Najork, col. 4, lines 59-64. According to that processing, the main procedure identifies URL's in the downloaded document that are candidates for downloading and processing (step 162). Najork, col. 4, line 66 - col. 5, line 3.

Najork specifically points out that "these URL's are typically found in hypertext links in the document being processed." Najork, col. 5, lines 3-4. But, as particularly pointed out in the present application, the present invention addresses the situation arising when URL's are *not* found in hypertext links, which presents a problem. That is, references from one web page to another may not be simply set out on the page as a straightforward hyperlink address [i.e., a URL], instead may be a script, form, selection menu, or button for example. Consequently, a conventional crawler and the crawler taught by Najork are not suitable for the "staticizing" problem addressed in the present invention. Present application, page 2, line 20 - page 3, line 6 (concluding, "Thus a need exists for improvements in crawler programs, to overcome their limitations so that they may be used for the staticizing problem as well as other applications."). Najork offers no teaching that addresses this problem, or even that suggests it exists.

The present application further elaborates on the problem, explaining how a reference that is not "simply set out on the page as a hyperlink address, but instead . . . specified by a script, for example, so that the address is produced only when a client browser executes the reference." Present application, page 5, lines 10 - 15; see also, page 12, line 21- page 13, line 1 (describing how the reference may be specified by a script, a selection menu, form, button or other element).

The present application goes on to explain how this problem may be addressed, as follows:

To generate references of this sort in connection with generating the requests to the server, another aspect of the invention arises. According to an embodiment, the crawler parses each received web page and sends references to an applet developed for an embodiment of the present invention that runs in the browser. (This applet may be referred to herein as a "JavaScript execution engine" or simply "JEE.") The browser determines the address for a next page responsive to such a reference, so that the browser may receive the next page and any cookie for the next page from the server, and the JEE returns the address and any cookie to the crawler program.

Present application, page 5, lines 15-22; page 15, lines 2-8. Accordingly, the subject claims state that the invention includes parsing a reference from one of the web pages by a crawler program and sending the reference to an applet running in a browser. The references do not teach or suggest this.

There is no evidence or suggestion in the cited art of determining the address for the next page by the browser executing the reference and sending the address to the crawler.

Neither Najork alone, nor Najork in combination with the other cited references, teach or suggest that a browser executes *a reference that has been sent to it by a crawler* in order to produce an address. Nor does Najork teach or suggest sending back to the crawler the address that results from this execution by the browser.

An example of a hypertext link in an HTML document that explicitly has the text of an address set out therein, as alluded to by Najork, is as follows: ``. The URL `"http://www.google.com/"` can be easily parsed from this link. In contrast, the following hypertext link provides an example of a reference that is not so straightforward and that is "specified by a script to produce an address" so that the address for the next page is determined "by the browser executing to the reference," as stated in amended claim 1 of the present application: ``. See Web page, http://freshair.npr.org/day_fa.jhtml?display=day&todayDate=09/21/2004. Executing this reference *produces* a URL with the standard hyperlink structure that includes "http://www" etc. The URL cannot be simply parsed from the `` reference.

The present application says "a reference . . . may be specified by a script" because an href tag, for example, typically has a *call* to a script and not the script *itself*. The browser locates the source code for the function that is called and then executes the specified function.

Note also, a "reference" is not limited to the context of an href tag. Consider the following example snippet of HTML code:

```
<form>  
<input type="button" value="GO" onclick="DoSearch()"/>  
</form>
```

This snippet creates a button that says “Go.” When the user presses the button the browser needs to execute the function DoSearch() in the context of the button before it can determine what URL to load. In this example also the URL to be loaded is “specified by a script,” the DoSearch() script, which is not itself included in the form that produces the button.

Note also the claims subject to the present ground of rejection, like the passage of the specification set out above, state that the browser executes the *reference* instead of saying merely that the browser executes the script. In the snippet example above, the crawler needs to know what URL to load when the button is pushed. The crawler achieves this by *telling the browser (via the applet) to push the button*. It cannot tell the browser to just execute the JavaScript function “DoSearch()” because the browser would then not have the context in which to execute the function. See present application, page 5, lines 10 - 15 (explaining that the address “is very dependent on the context in which it is produced, that is, the history that led up to it, including the state of the server and the client browser.”); see also, page 15, lines 2-8 (explaining that the crawler passes information 230 to a JavaScript execution engine 210 for generating queries to the web server 100 and that the information includes the JavaScript command that invokes script 303 when button 304 is clicked, a context object, the browser window object, and the document object associated with page 140.X in its context as it exists, loaded in browser 205).

Appellant recognizes that *executing* a reference to produce an address, as claimed, might be confused with *parsing* the reference to find an address that is explicitly set out therein. The explanation above clarifies these significant differences. Also, to make the distinction particularly clear in the claims, Appellant previously amended claim 1, to particularly point out that “a page has a reference, wherein the reference is specified by a script for producing an address for a next page” and that “such a reference is parsed from one of the web pages by the crawler program and sent to an applet running in a browser.” Claims 9 and 17 were previously amended to include similar language. Further, the claims were amended to clearly tie different aspects together by stating that the address for the next page is determined by the browser “executing” the reference and sending the address to the crawler.

It should be clear from the discussion above that claims 1, 9 and 17 are patentably distinct from the teaching of Najork that the Office action relies upon for the rejection. For these reasons Appellant contends that claims 1, 9 and 17 are allowable.

In reply to the above arguments regarding the first ground of rejection, the present Office action, page 3, points out that Najork teaches “identifying a reference (URL) for the next page to be downloaded by executing thread 130 located in the web crawler” and that Kredo teaches that a reference may be specified by a script. On this basis the present Office action asserts that it is obvious to “determin[e] the [address] for the next page by the browser executing the reference and sending the address to the crawler,” as claimed. However, as Appellant has repeatedly pointed out, neither Najork alone, nor Najork in combination with Kredo or the other cited references, teach or suggest that a browser executes *a reference that has been sent to it by a crawler* in order to produce an address, nor sending back to the crawler the address that results from this execution by the browser.

2. The rejection of claims 2, 10 and 18 under 35 U.S.C. 103(a) as being unpatentable over Najork, Kredo and US Patent 6,735,169 (“Albert”) is not proper because there is no evidence or suggestion in the cited art of a proxy and a web site server have different IP addresses, but a resolver file indicates they are the same.

The subject claims clearly state that a proxy and a web site server have different IP addresses, but a resolver file indicates they are the same.⁴ See, for example, claim 2 (stating that the browser is configured to use a certain proxy and refer to a resolver file for hostname-to-IP-address-resolution, that the resolver file indicates the IP address of the proxy *is* the IP address for the web site server, and that the IP address of the proxy *is not* the IP address of a web site server queried by the crawler program). Claims 10 and 18 have similar language. Appellant respectfully submits that there is no evidence or suggestion in the cited art of all the limitations of the subject claims and that the rejection is, therefore, improper. “[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support

⁴ Because of this intentional inconsistency, the specification calls the proxy gateway a “spoof proxy.” Present application, page 6, lines 13-14.

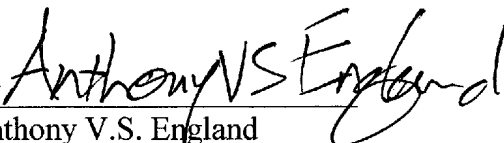
the legal conclusion of obviousness.” In re Kahn, 441 F. 3d 977, 988 (CA Fed. 2006) cited with approval in KSR Int’l v. Teleflex Inc., 127 S. Ct. 1727, 82 USPQ2d 1385 (2007). All claim limitations must be considered MPEP 2143.03 (citing In re Wilson, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970)).

The rejection relies on FIG. 3A in Albert, which shows a forwarding agent 302 between a client 304 and a virtual machine 310. See also Albert et al., col. 11, line 33, through col. 12, line 37. However, this does not teach or suggest an arrangement, as claimed, in which the client browser is configured to use a certain proxy and refer to a resolver file for hostname-to-IP-address-resolution, where the proxy and the web site server queried by the crawler program have *different* IP addresses but the resolver file indicates they are the *same*.

REQUEST FOR ACTION

For the above reasons, Appellant requests that all the pending claims of the present application be allowed and that the application be promptly passed to issuance.

Respectfully submitted,

By 
Anthony V.S. England
Registration No. 35,129
Attorney of Record for
IBM Corporation
Telephone: 512-477-7165
a@aengland.com

Attachments: Claims Appendix, Evidence Appendix, Related Proceedings Appendix

APPENDIX "AA" CLAIMS

1. (previously presented) A method for crawling a web site, the method comprising the steps of:

a) querying a web site server by a crawler program, wherein at least one page of the web site has a reference, wherein the reference is specified by a script to produce an address for a next page;

b) parsing such a reference from one of the web pages by the crawler program and sending the reference to an applet running in a browser; and

c) determining the address for the next page by the browser executing the reference and sending the address to the crawler.

2. (previously presented) The method of claim 1, the browser being configured to use a certain proxy and refer to a resolver file for hostname-to-IP-address-resolution, wherein the web site server has an IP address and the proxy for the browser has a certain IP address, the certain IP address of the proxy being different than the IP address of the web site server, and wherein the resolver file indicates the certain IP address of the proxy as the IP address for the web site server.

3. (original) The method of claim 2, comprising the steps of:
adding an onload attribute to one of the web pages by the proxy;
defining an event handler for the onload attribute by the proxy, wherein the event handler sets a certain variable; and
polling the certain variable by the applet to determine when the page is loaded.

4. (previously presented) The method of claim 1, wherein the crawler is programmable to perform particular action sequences for selecting non-hypertext-link parameters from the at least one web page in a particular sequence, so that the queries to the web server include the selected parameters and a context arising from the particular sequence.

APPENDIX "AA" CLAIMS

5. (previously presented) The method of claim 1, at least one of the web pages being dynamically generated by the server responsive to corresponding ones of the queries, comprising the step of:

processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server.

6. (original) The method of claim 5, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, and wherein the step of processing the server generated web pages comprises the step of:

removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

7. (previously presented) A method for reducing dynamic data generation on a web site server, the method comprising the steps of:

a) querying a web site server by a crawler program responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web pages being dynamically generated; and

b) processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, the operation including a number of non-hypertext-link elements on the first page selected in a particular sequence, and wherein the step of processing the server generated web pages comprises the step of:

removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

APPENDIX "AA" CLAIMS

8. (canceled)

9. (previously presented) A computer program product for crawling a web site, wherein the computer program product resides on a computer usable medium having computer readable program code, the program code comprising:

a) first instructions for querying a web site server by a crawler program, wherein at least one page of the web site has a reference, wherein the reference is specified by a script for producing an address for a next page;

b) second instructions for parsing such a reference from one of the web pages by the crawler program and sending the reference to an applet running in a browser; and

c) third instructions for determining the address for the next page by the browser executing the reference and sending the address to the crawler.

10. (previously presented) The computer program product of claim 9, the browser being configured to use a certain proxy, and refer to a resolver file for hostname-to-IP-address-resolution, wherein the web site server has an IP address and the proxy for the browser has a certain IP address, the certain IP address of the proxy being different than the IP address of the web site server, and wherein the resolver file indicates the certain IP address of the proxy as the IP address for the web site server.

11. (original) The computer program product of claim 10, comprising:
fourth instructions for adding an onload attribute to one of the web pages by the proxy;
fifth instructions for defining an event handler for the onload attribute by the proxy;
wherein the event handler sets a certain variable; and
sixth instructions for polling the certain variable by the applet to determine when the page is loaded.

APPENDIX "AA" CLAIMS

12. (previously presented) The computer program product of claim 9, wherein the first instructions comprise instructions for causing the crawler to perform particular action sequences for selecting non-hypertext-link parameters from the at least one web page in a particular sequence, so that the queries to the web server include the selected parameters and a context arising from the particular sequence.

13. (previously presented) The computer program product of claim 9, at least one of the web pages being dynamically generated by the server responsive to corresponding ones of the queries, the computer program product comprising:

instructions for processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server.

14. (original) The computer program product of claim 13, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, and wherein the instructions for processing the server generated web pages to generate corresponding processed versions of the web pages comprise:

instructions for removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

15. (previously presented) A computer program product for reducing dynamic data generation on a web site server, wherein the computer program product resides on a computer usable medium having computer readable program code, the program code comprising:

first instructions for querying a web site server by a crawler program responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web pages being dynamically generated; and

APPENDIX "AA" CLAIMS

second instructions for processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, the operation including a number of non-hypertext-link elements on the first page selected in a particular sequence, and wherein the second instructions comprise:

instructions for removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

16. (canceled)

17. (previously presented) An apparatus for crawling a web site, the apparatus comprising:

a processor connected a network,

a storage device connected to the processor and the network, wherein the storage device is for storing a program for controlling the processor, and wherein the processor is operative with the program to execute a crawler program and a browser program for performing the steps of:

a) querying a web site server by the crawler, wherein at least one page of the web site has a reference, wherein the reference is specified by a script for producing an address for a next page;

b) parsing such a reference from one of the web pages and sending the reference to an applet running in a browser; and

c) determining the address for the next page by the browser executing the reference and sending the address to the crawler.

APPENDIX "AA" CLAIMS

18. (previously presented) The apparatus of claim 17, the browser being configured to use a certain proxy and refer to a resolver file for hostname-to-IP-address-resolution, wherein the web site server has an IP address and the proxy for the browser has a certain IP address, the certain IP address of the proxy being different than the IP address of the web site server, and wherein the resolver file indicates the certain IP address of the proxy as the IP address for the web site server.

19. (original) The apparatus of claim 18, wherein an onload attribute is added to one of the web pages by the proxy, and an event handler is defined for the onload attribute to set a certain variable, and wherein the processor is operative with the program for performing the step of:

polling the certain variable by the applet to determine when the page is loaded.

20. (previously presented) The apparatus of claim 17, wherein the processor is operative with the program for causing the crawler to perform particular action sequences for selecting non-hypertext-link parameters from the at least one web page in a particular sequence, so that the queries to the web server include the selected parameters and a context arising from the particular sequence.

21. (previously presented) The apparatus of claim 17, at least one of the web pages being dynamically generated by the server responsive to corresponding ones of the queries, wherein the processor is operative with the program for performing the step of:

processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server.

22. (original) The apparatus of claim 21, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a

APPENDIX "AA" CLAIMS

second web page if the first page were used to generate further requests to the server, and wherein the step of processing the server generated web pages comprises the step of:

removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

23. (previously presented) An apparatus for reducing dynamic data generation on a web site server, the apparatus comprising:

a processor connected to a network,

a storage device connected to the processor and the network, wherein the storage device is for storing a program for controlling the processor, and wherein the processor is operative with the program to execute a crawler program and a browser program for performing the steps of:

a) querying a web site server by the crawler responsive to references from one web page to another in the web site, wherein the queries are for causing the server to generate web pages, at least one of the web pages being dynamically generated; and

b) processing the server generated web pages to generate corresponding processed versions of the web pages, so that the processed versions can be served in response to future queries, reducing dynamic generation of web pages by the server, wherein at least a first such server generated web page has included in it an operation that would cause the server to dynamically generate a second web page if the first page were used to generate further requests to the server, the operation including a number of non-hypertext-link elements on the first page selected in a particular sequence, and wherein the step of processing the server generated web pages comprises the step of:

removing the operation from the first server generated web page and replacing the operation with a reference to a version of another of the server generated web pages.

24. (canceled)

APPENDIX "BB" EVIDENCE

NONE.

APPENDIX "CC" RELATED PROCEEDINGS

NONE.